Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method of operating a multithreaded parallel processor comprising:

directing the processor having a plurality of microengines to swap, based on a ~~voluntary swap~~ user-specified parameter specified in a context-swap instruction, a currently running context, corresponding to a first thread, in a specified microengine to let another context, corresponding to a different thread that is ready to execute, execute in that microengine and cause a different context and associated program counter to be selected, with the swapped first thread automatically re-enabled to run at some subsequent context arbitration point, ~~and~~

wherein directing the processor comprises waking up the swapped out context when the ~~voluntary swap~~ user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event.

2-3. (Cancelled)

4. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "sram Swap", which swaps out the current context and wakes up the swapped, current context when the thread's SRAM signal is received.

5. (Currently amended) The method of claim 1 wherein the ~~voluntary swap~~ user-specified parameter specifies "sdram Swap," which swaps out the current context and wakes up the swapped, current context when the thread's SDRAM signal is received.

6. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "FBI" which swaps out the current context and wakes up the swapped, current context when the thread's FBI signal is received indicating that an FBI CSR, Scratchpad, TFIFO, or RFIFO operation has completed.

7. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "seq_num1_change/seq_num2_change", which swaps out the current context and wakes up the swapped, current context when a value of the sequence number changes.

8. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "inter_thread" which swaps out the current context and wakes up the swapped, current context when the thread's interthread signal is received.

9. (Cancelled)

10. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "auto_push" which swaps out the current context and wakes up the swapped, current context when SRAM transfer read register data has been automatically pushed by a FBus interface.

11. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "start_receive" which swaps out the current context and wakes up the swapped, current context when new data in a receive FIFO is available for this thread to process.

12. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "kill" which prevents the current context or thread from executing again until an appropriate enable bit for the thread is set in a CTX_ENABLES register.

13. (Currently amended) The method of claim 1 wherein the user-specified ~~voluntary swap~~ parameter specifies "pci" which swaps out the current context and wakes up the swapped, current context when a PCI unit signals that a DMA transfer has been completed.

14. (Previously presented) The method of claim 1 wherein directing further comprises:

in response to an optional token "defer one" specified in the context-swap instruction, executing an additional instruction in an instruction stream of the currently running context before the context is swapped.

15. (Currently amended) A method of operating a multithreaded parallel processor, the method comprising:

receiving ~~an indication of a voluntary swap, the voluntary swap specified as~~ a user-specified parameter specified in a context-swap instruction;

performing, ~~in response to the received indication,~~ a swapping operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, the swapped first thread being automatically re-enabled to run at some subsequent context arbitration point; and

waking up the swapped out context when the ~~voluntary swap~~ user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event.

16. (Previously presented) The method of claim 15 wherein performing comprises swapping a currently running context in a specified microengine to let another context execute in that microengine.

17. (Cancelled)

18. (Currently amended) The method of claim 15 wherein the user-specified ~~voluntary swap~~ parameter specifies "sram Swap", and performing a swapping comprises swapping out the current context and waking up the swapped, current context when the thread's SRAM signal is received.

19. (Currently amended) The method of claim 15 wherein the user-specified ~~voluntary swap~~ parameter specifies "sram Swap", and performing a swapping comprises swapping the current context and waking up the swapped, current context when the thread's SDRAM signal is received.

20. (Currently amended) The method of claim 15 wherein the user-specified ~~voluntary swap~~ parameter specifies "inter_thread" which swaps out the current context and wakes up the swapped, current context when the thread's interthread signal is received.

21. (Previously presented) The method of claim 15 further comprising:

in response to an optional token "defer one" specified in the context-swap instruction, executing an additional instruction in an instruction stream of the currently running context before the context is swapped.

22. (Currently amended) A parallel processor that can execute multiple contexts and that comprises:

a register stack;

a program counter for each executing context;

an arithmetic logic unit coupled to the register stack and a program control store that stores a context swap instruction that causes the processor to:

receive ~~an indication of a voluntary swap, specified as~~ a user-specified parameter specified in the context swap instruction;

perform, ~~in response to the received indication,~~ a swap operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, the

swapped first thread is automatically re-enabled to run at some subsequent context arbitration point; and

wake up the swapped out context when the ~~voluntary swap~~ user-specified parameter specified in the context-swap instruction is activated, the user-specified parameter specifying an occurrence of an event.

23. (Cancelled)

24. (Currently amended) A computer program product residing on a computer readable storage device ~~medium~~ for causing a multithreaded parallel processor to perform a function, the computer program product comprising instructions causing the processor to:

receive ~~an indication of a voluntary swap, specified as~~ a user-specified parameter specified in a context-swap instruction;

perform, ~~in response to the received indication,~~ a swapping operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, the swapped first thread is automatically re-enabled to run at some subsequent context arbitration point; and

wake up the swapped out context when the ~~voluntary swap~~ user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event.

25. (Cancelled)

26. (New) The method of claim 1 wherein the user-specified parameter specifies "voluntary".